

## ARBORI

Arborii sunt un caz particular al grafurilor (cele mai simple ca structură și cele mai utilizate în practică).

**Arbore** = Graf neorientat, **conex** și **fără cicluri**.

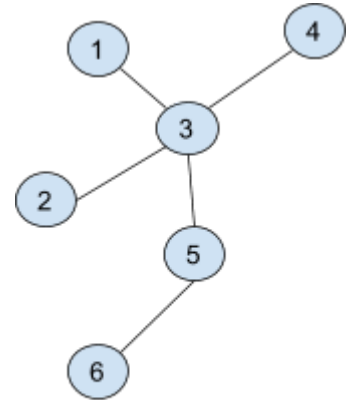
(sau) Graf **fără cicluri** în care  $m = n - 1$ , unde  $m$  = numărul de muchii;  $n$  = numărul de noduri.

(sau) Graf **conex** în care  $m = n - 1$ , unde  $m$  = numărul de muchii;  $n$  = numărul de noduri.

(sau) Graf fără cicluri, maximal (orice muchie adăugată în plus formează un ciclu).

(sau) Graf conex, minimal (dacă se șterge vreuna din muchiile existente se pierde conexitatea).

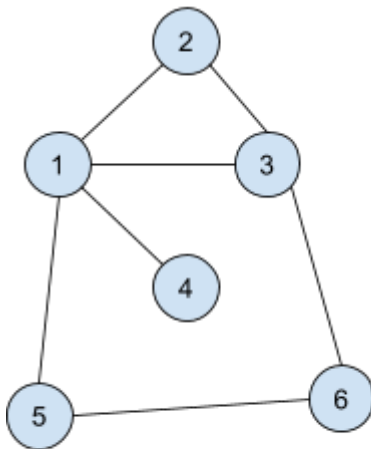
(sau) Graf în care orice pereche de noduri este legată de un lanț și numai unul.



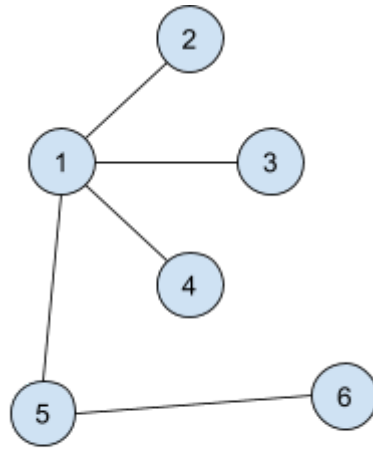
Exemplu de arbore

**Arbore parțial** al unui graf neorientat  $G$  = un graf parțial al lui  $G$  care îndeplinește proprietatea de a fi *arbore*.

Obs. Un graf neorientat conține un arbore parțial dacă și numai dacă este *conex*.



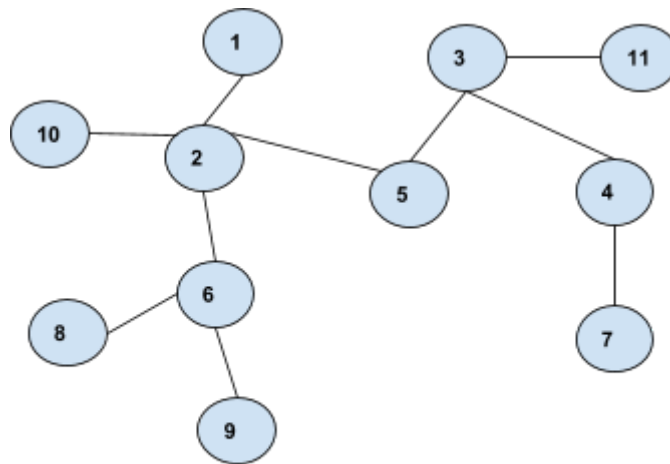
graf neorientat conex  $G$



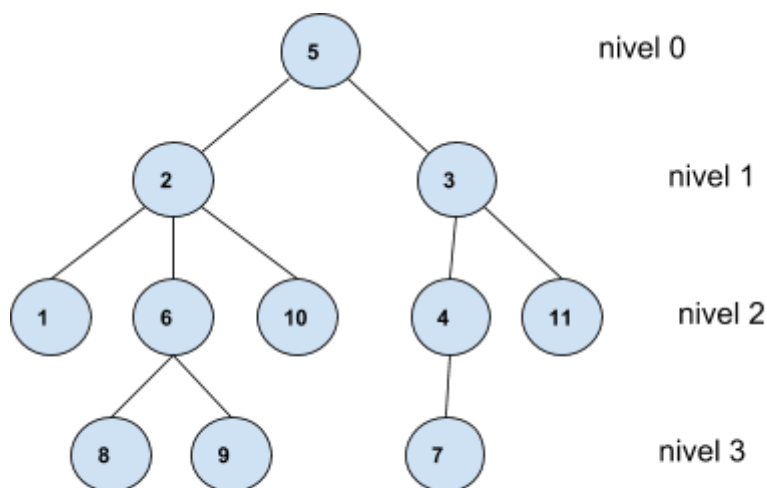
un arbore parțial al grafului  $G$

**Arbore cu rădăcină** = un arbore în care unul dintre noduri este desemnat rădăcină (aceasta determină așezarea nodurilor pe nivele).

**Rădăcină** = nod special care generează așezarea nodurilor arborelui pe nivele. Nivelul unui nod se determină calculând lungimea lanțului care îl leagă de rădăcină. Rădăcina se află pe nivelul 0.



arbore fără rădăcină



arbore cu rădăcină

În exemplul de mai sus, rădăcina este nodul 5 și se află pe nivelul 0. Iar pe nivelul 2 se află nodurile: 1, 4, 6, 10 și 11.

Algoritmii ce folosesc arbori sunt ușor de implementat recursiv deoarece la fiecare pas putem separa arborele în mai mulți subarbori.

Putem spune că: **un arbore** este format dintr-o rădăcină la care sunt conectați un număr finit de arbori (**definiție recursivă arbore**).

## Noțiuni din terminologia arborilor cu rădăcină:

**Descendent (urmaș)** = într-un arbore cu rădăcină un nod x este descendentul (urmașul) unui nod y dacă există un lanț care le unește și nu trece prin rădăcină și x se află pe un nivel mai mare decât y (sub y).

*De exemplu:* în figura de mai sus, descendenții sau urmașii lui 2 sunt: 1, 6, 10, 8 și 9, iar descendenții lui 5 sunt restul nodurilor din arbore (deoarece 5 e rădăcina).

**Descendent direct (fiu)** = într-un arbore cu rădăcină un nod x este descendentul direct (fiul) unui nod y dacă există muchie între ele și nivelul lui x este mai mare decât al lui y.

*De exemplu:* în figura de mai sus, descendenții direcți (fiii) lui 2 sunt: 1, 6 și 10.

**Ascendent (strămos)** = într-un arbore cu rădăcină un nod x este ascendentul (strămoșul) unui nod y dacă există un lanț care le unește și nu trece prin rădăcină și x se află pe un nivel mai mic decât y (deasupra lui y). Atenție! Lanțul respectiv este format din noduri de pe nivele diferite.

*De exemplu:* în figura de mai sus, ascendenții lui 8 sunt: 6, 2 și 5 (atenție nu și 1).

**Ascendent direct (tată)** = într-un arbore cu rădăcină un nod x este ascendentul direct (tatăl) unui nod y dacă există muchie între ele și nivelul lui x este mai mic decât al lui y.

*De exemplu:* în figura de mai sus, ascendentul direct (tata) lui 8 este 6.

**Frați** = noduri care au același tată.

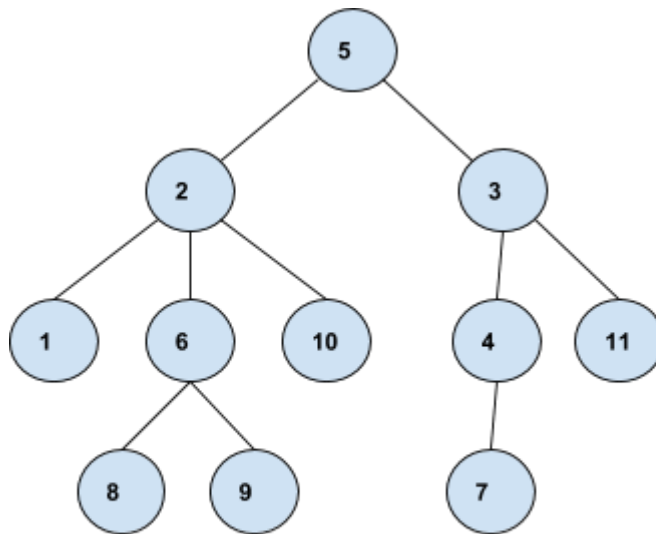
*De exemplu:* în figura de mai sus, 1, 6 și 10 sunt frați (au același tată).

**Frunze (noduri terminale)** = nodurile care nu au descendenți.

*De exemplu:* în figura de mai sus, frunzele sunt: 1, 8, 9, 10, 7 și 11.

## Metode de reprezentare/memorare a arborilor

- ❑ **Matrice de adiacență** - se procedează la fel ca la un graf normal
- ❑ **Liste de adiacență (de descendenți)** - se procedează la fel ca la listele de adiacență pentru graf normal dar trebuie să se cunoască în plus rădăcina, iar în listele de adiacență se rețin doar descendenții direcți (fiii) ai unui nod
- ❑ **Vectorul de tați** - trebuie să cunoaștem și rădăcina. Într-un vector, pe poziția corespunzătoare fiecărui nod se memorează tata nodului respectiv, excepție pentru rădăcină, pe poziția corespunzătoare acesteia se memorează valoarea 0.



**Exemplu.** - pentru arborele cu rădăcină de mai sus, **vectorul de tați** este:

2	5	5	3	0	2	4	6	6	2	3
1	2	3	4	5	6	7	8	9	10	11

Surse:

*Terminologie grafuri - Eugen Neamțiu*

[https://ro.wikipedia.org/wiki/Arbore\\_\(teoria\\_grafurilor\)](https://ro.wikipedia.org/wiki/Arbore_(teoria_grafurilor))

[https://ro.wikipedia.org/wiki/Arbore\\_par%C8%9Bial](https://ro.wikipedia.org/wiki/Arbore_par%C8%9Bial)

<https://invata.info/2017/04/02/arbori-c/>

<http://infoscience.3x.ro/c++/Arbori.htm>

Alte surse utile:

<https://olidej.wikispaces.com/file/view/Arbori+partiali.pdf>

[http://campion.edu.ro/arhiva/www/arhiva\\_2009/seds/14/index.htm](http://campion.edu.ro/arhiva/www/arhiva_2009/seds/14/index.htm)